Reproduction of Genome Correction Software

Georgia Tech College of Computing

Arav Thadani

Spring 2019

Abstract

The primary issues with genome sequencing machines today are insertion, substitution, and deletion. These have led to the rise of genome correction software which use different algorithms to correct errors in the sequence. The purpose of this study is to test around 12 of the most popular genome correcting software and see how the results that we obtain compare to the results that are reported. We use Nextflow as the pipeline software and Docker containers so that the environment remains constant and can be replicated by anyone after us to see our results. Each testing case started off with a Docker container where we pre-install the correction software along with indexing software. Then we move on to the Nextflow template that consists of the datasets that we will be testing. The next section is the primary indexing followed by running the actual correction software on the dataset. Lastly, we have to do another round of indexing and then final measure results by running a script which tells us how many well the software ran. The testing programs are usually custom python scripts that output in the format provided in the correction software's paper. We have published a website which feature all the results that we have found. Within the website, results are divided up by software. Within each software, one can see the results we found next to the results that are published and the discrepancy between the two.

Introduction

With the rise of diseases in the world today, there has been a lot more research and manpower put to using technology to help with the field of bioinofrmatics. Despite all the progress, there are still some major issues that we face today. Multiple systems exist these days

that can sequence large amounts of data at faster speeds and lower costs. One issue that continues to persist though are the errors that are produced by these systems when they sequence. The primary errors that are produced when sequencing are insertion, deletion, and substitution. In order to properly study the genome, scientists try to get as close to zero errors as possible as a perfect genome would help make the most progress for research. Sequencing machines each perform their best with specific data sets of specific sizes. Although the varying sizes of input might help to reduce occurrences of errors, it does not fully eliminate them (1). Genome correction software were built because of this particular issue.

Currently in the market there are several different developers out there which each boast that their software does the best job of correcting the errors stated before. Similar to the machines that do the sequencing, there is specific indexed data from certain machines that each software claims it works best for. There are several different algorithms used by these software which commonly involve recursion and k-mers for corrections (7). For the purpose of this project, we focused much more on the testing methods than the actual logic used in correcting. The goal of this project is to reproduce the results of several of the most popular software out there. We want to see how similar our results are to those of the ones published by the papers we are analyzing. In addition, we would also like to see which software works best for different variables and conditions of the dataset.

The most time-consuming aspect of this project is the fact that each software comes with its own programs it needs to run, different software it uses for indexing, and then different methods it uses to test against the genome. In order to do this, we will be creating Docker containers where we will pre-install each API or program needed for testing. We will also be

using NextFlow scripts as a pipelining software which streamlines the testing process. This also means that we have to custom make each one of these containers and scripts for each software we decide to test. In this paper we will reports the results we have found and also let each creator of the software know what we found and if there is a discrepancy in what they reported.

Literature Review

One of the foundations of modern biology is the ability to sequence genomes (4). These genomes could be those of bacteria like e. coli or genomes of humans themselves. Each one in the hope they are able to relay to us more information about life and how to make it better. The process is a multi-faceted one that requires tools like assemblers and aligners. These tools take in the data and then output a more readable version. It is naturally understandable then that when there are errors in the data being input there are going to be errors in the output. This is what sheds light on the need for error correction tools (5). When we allow these mistakes to occur in the real world and in experiments it can have huge implications such as erroneous genome assemblies and mistaken understanding regarding RNA/DNA editing (4).

First generational error correcting genome software focused more on the algorithm and accuracy of the results. The software that we will be reproducing in our experiment is primarily second generational and focuses much more on optimization regarding large datasets. The goal now is to be able to produce accurate reads when given a large dataset in an efficient time manner (5 and 6). To understand how necessary the software is we must be made aware of how wide spread the problem is. Illumina which is the most popular machine for sequencing

genomes has errors in half of its reads, on average (2). There are several dozen software out their today that do their best to try an solve this problem but there are only around a couple dozen which are the most popular that do well with accuracy, space efficiency, and time efficiency. Each one of these software have a paper which tell us more about what strategy they are using, the datasets they have tested, and the results they have gotten. Papers like Quake and Racer give us comparisons of same datasets and how they perform with competing software (2 and 3). Often times the software reports that theirs is performing the best and providing the most accurate results. Since so much of the technology we are testing is so new, there is not much analysis on them from anyone other than those whose technology it is and those are who competing against it. This leaves a gap in the research field that we hope to fill. The goal of our experiment is to come in as a third unbiased party which can reproduce each experiment and report our results compared to what the paper reports.

The primary form of assembly that these papers utilize are de novo assembly which just means that the sequence is corrected without the reference genome (3). This can similarly be compared to in sample and out of sample testing. Within de novo, there are several different approaches that can be taken, but the most popular is using k-mers (7). The umbrella under k-mers is much larger but it is important because we also hope to see a correlation in how accurate results are in relation to the algorithm that was used for correction. Current studies in the field focus much more on developing newer, more accurate and efficient algorithms. Even though this is critical, there is very little study going on regarding actual analysis of software results. In addition to reporting the results our studies achieve, we also plan on reaching out to the authors of papers and letting them know of any discrepancies we have found. This is not at

all to discredit his work but more to spark a discussion on how our implementations might have differed. In conclusion, we hope our research gives people in the field of bioinformatics an unbiased view on how different software performs on similar datasets because there is no source for this type of information out right now.

Methods

The purpose of the experiment was to produce a program which can test the genome correction software and output its accuracy. When starting the reproducibility experiment, the researchers began by reading through the paper of the genome correction software that is being tested and gather the datasets needed, the software that was used for indexing, how to obtain the software for correction, and then also how the results were judged. With this information, they began writing the testing program. They stated with a Docker container and wrote the terminal line commands to install the indexing and correction software necessary. This can usually be done by using the commands “wget” and “git clone” following the link for the download. Often the software for correction might need pre-installments itself to run so that required reading the documentation and then going about finding out how to download the pre-installments for that. While writing the Docker container, it was important to test after each line is added to make sure the correct item is being downloaded and is also being placed in the right location.

Once the Docker was finished and correctly with all the commands for indexing and correcting, one can move on to the Nextflow file where the pipelining began. The first step in the Nexflow document was to download the datasets that are being tested. These can usually

be found as public information. The other key component was the correctly formed genome that was used later as comparison to see the errors made and resolved. Once the datasets and their matching correct genome were downloaded, the researchers moved onto the indexing. Indexing is usually done by two primary programs: BWA or Bowtie2. If not by these two, then usually by a version of one or the other and sometimes even a combination of both. It was important to first build the indexed file before applying the correction software to make it easier to process.

The next step was the primary one which is adding code to actual correct software. Usually there were certain parameters required such as how large the genome was or in the k-mer correction, a popular algorithm used to make corrections, the specific k value desired. When actual testing, this was the part that usually takes the longest. Once the actual correction aspect is finished, the researchers moved on to indexing again. The reason for the indexer in this situation was to align the newly corrected reads along with the correct genome. This way, when calculating how efficient the correction software was, it was much easier to process. The final step was to build a custom python script which takes the newly indexed data and does a comparison. This script had to be custom because each software judges its accuracy in different way. Some software might see how many new errors are created while others see how many errors were resolved and some might do a combination of both. At the end of the python script a new page was created outputting the results that have been calculated. Like the Docker program, it was important to test after each step to make sure it is working correctly. Keep in mind the researchers emphasized to run within the Docker container because that is where the

programs are stored. Once the whole Nextflow document is prepared, there was an outputted page stating the results found that could be compared to what the paper reported.

Results

For this particular reproducibility project, results can be measured by comparing how similar the results are between the ones that were computed versus the ones that were reported by the authors of the papers of software. Thus, for each software, the researchers have displayed the metrics computed, the metrics reported, and the differences between the two. Since the experiment is still in progress, the only software that we do have results for are the following four: Racer, Lighter, BLUE, and Trowel. Each software has its own metrics that it uses to measure accuracy.

Racer is a rare software because it only uses gain as a measure of its success. There is a total of 15 organisms that were tested but what makes it interesting is that they were once tested using the indexer BWA and the other time they were tested using Read Search. The results are as following:

RACER evaluation using Read Search

Search:

Organism	Dataset	Gain (Computed)	Gain (Reported)	Difference
B. subtilis subtilis 168	DRR000852	83.64	82.12	1.52
C. elegans WS222	SRR065390	58.54	56.54	2.00
D. melanogaster R6.18	SRR018292,SRR018293,SRR018294,SRR060098	38.85	42.95	-4.10
E. coli K 12 MG1655	SRR001665	90.78	86.32	4.46
E. coli K 12 MG1655	SRR022918	39.26	56.50	-17.24
E. coli K 12 MG1655	SRR396532	80.29	76.32	3.97
E. coli K 12 MG1655	SRR396536	86.78	83.58	3.20
H. influenzae Rd KW20	SRR065202	76.77	78.35	-1.58
L. interrogans str. 56601	SRR353563	90.97	53.91	37.06
L. interrogans str. Fiocruz L1 130	SRR397962	88.14	59.87	28.27
L. lactis	SRR088759	97.63	80.49	17.14
P. aeruginosa PAO1	SRR396641	86.73	85.32	1.41
S. aureus MW2	SRR022866	26.89	25.96	0.93
S. cerevisiae S288C	SRR352384	13.59	12.25	1.34
T. pallidum	SRR361468	78.77	85.75	-6.98

RACER evaluation with BWA

Search:

Organism	Dataset	Gain (Computed)	Gain (Reported)	Difference
B. subtilis subtilis 168	DRR000852	26.26	93.55	-67.29
C. elegans WS222	SRR065390	9.29	65.96	-56.67
D. melanogaster R6.18	SRR018292,SRR018293,SRR018294,SRR060098	28.15	57.04	-28.89
E. coli K 12 MG1655	SRR001665	78.44	82.67	-4.23
E. coli K 12 MG1655	SRR022918	31.18	90.80	-59.62
E. coli K 12 MG1655	SRR396532	56.99	83.91	-26.92
E. coli K 12 MG1655	SRR396536	85.23	77.80	7.43
H. influenzae Rd KW20	SRR065202	50.85	84.33	-33.48
L. interrogans str. 56601	SRR353563	34.88	89.27	-54.39
L. interrogans str. Fiocruz L1 130	SRR397962	29.09	90.81	-61.72
L. lactis	SRR088759	95.93	92.02	3.91
P. aeruginosa PAO1	SRR396641	48.34	89.31	-40.97
S. aureus MW2	SRR022866	12.25	47.00	-34.75
S. cerevisiae S288C	SRR352384	8.35	14.68	-6.33
T. pallidum	SRR361468	77.42	92.35	-14.93

The next software to analyze is Lighter. Lighter also uses a single metric but instead of gain, they use percent increase. Lighter tests three organism under two conditions, one with an individual read comparison and one with a base comparison.

Lighter bases

Search:

Organism	Dataset	Pct Increase (Computed)	Pct Increase (Reported)	Difference
C. elegans WS222	SRR065390	0.62	0.43	0.19
E. coli K12 MG1655	ERR022075	1.14	0.61	0.53
H. sapiens Chr 14	GAGE	1.67	0.74	0.93

Showing 1 to 3 of 3 entries

Lighter reads

Search:

Organism	Dataset	Pct Increase (Computed)	Pct Increase (Reported)	Difference
C. elegans WS222	SRR065390	0.21	0.10	0.11
E. coli K12 MG1655	ERR022075	0.52	0.42	0.10
H. sapiens Chr 14	GAGE	0.93	0.91	0.02

Showing 1 to 3 of 3 entries

Blue is very unique because it has the most amount of metrics to evaluate its success. Blue checks how many reads with zero to ten edits were there before the correction software was run and then how many reads with zero to ten edits were there afterwards. This allows much of the interpretation up to the reader. There were three organisms run but only two are shown because the other is of the same organism as the E. Coli displayed.

Dataset : ERR330008 (P. aeruginosa)

Search:

	Stat	Before	Before (reported)	Differnce	After	After (reported)	Difference
1	No. Total Reads	10020458	9859280	161178	10020458	9859280	161178
2	No. Perfect eads	6993836	9728163	-2734327	9878651	9728163	150488
3	No. Reads w. 1 edit	1193434	1170891	22543	15573	8327	7246
4	No. Reads w. 2 edits	451595	445059	6536	2450	2302	148
5	No. Reads w. 3 edits	260289	257066	3223	1218	833	385
6	No. Reads w. 4 edits	176019	174212	1807	472	542	-70
7	No. Reads w. 5 edits	128343	127036	1307	511	451	60
8	No. Reads w. 6 edits	99270	98281	989	480	443	37
9	No. Reads w. 7 edits	79361	78481	880	624	410	214
10	No. Reads w. 8 edits	65818	65144	674	478	467	11
11	No. Reads w. 9 edits	54780	53789	991	596	390	206
12	No. Reads w. 10+ edits	517713	402036	115677	119405	30533	88872
13	% Perfect Reads	69.796	69.83	-0.034	98.584	98.67	-0.086
14	% Reads w. 1 edit	11.909	11.88	0.029	00.155	0.08	0.075
15	% Reads w. 2 edits	4.506	4.51	-0.004	00.024	0.02	0.004
16	% Reads w. 3 edits	2.598	2.61	-0.012	00.012	0.01	0.002
17	% Reads w. 4 edits	1.756	1.77	-0.014	00.004	0.01	-0.096
18	% Reads w. 5 edits	1.281	1.29	-0.009	00.005	0.00	0.005
19	% Reads w. 6 edits	0.991	1.00	-0.009	00.004	0.00	0.004
20	% Reads w. 7 edits	0.792	0.80	-0.008	00.006	0.00	0.006
21	% Reads w. 8 edits	0.657	0.66	-0.003	00.004	0.00	0.004
22	% Reads w. 9 edits	0.547	0.55	-0.003	00.005	0.00	0.005
23	% Reads w. 10+ edits	5.17	4.08	1.09	1.19	0.31	0.88

Dataset : SRR029323 (E. coli)

Search:

	Stat	Before	Before (Reported)	Difference	After	After (Reported)	Difference
1	No. Total Reads	287672	143836	143836	287672	143836	143836
2	No. Perfect Reads	174617	1092	173525	184302	73135	111167
3	No. Reads w. 1 edit	21233	7552	13681	20830	26115	-5285
4	No. Reads w. 2 edits	18399	16113	2286	15963	13895	2068
5	No. Reads w. 3 edits	12521	27488	-14967	10960	10926	34
6	No. Reads w. 4 edits	8230	24044	-15814	7145	3705	3440
7	No. Reads w. 5 edits	5889	17092	-11203	5196	1056	4140
8	No. Reads w. 6 edits	4474	10997	-6523	3984	806	3178
9	No. Reads w. 7 edits	3726	6759	-3033	3257	598	2659
10	No. Reads w. 8 edits	2952	4475	-1523	2612	509	2103
11	No. Reads w. 9 edits	2533	3167	-634	2225	394	1831
12	No. Reads w. 10+ edits	33098	15144	17954	31198	5673	25525
13	% Perfect Reads	60.70	0.8	59.9	64.06	50.8	13.26
14	% Reads w. 1 edit	7.38	5.0	2.38	7.24	18.2	-10.96
15	% Reads w. 2 edits	6.40	11.2	-4.8	5.54	9.7	-4.16
16	% Reads w. 3 edits	4.35	19.1	-14.75	3.80	7.6	-3.8
17	% Reads w. 4 edits	2.86	16.7	-13.84	2.48	2.6	-0.12
18	% Reads w. 5 edits	2.05	11.9	-9.85	1.80	0.7	1.1
19	% Reads w. 6 edits	1.56	7.6	-6.04	1.38	0.6	0.78
20	% Reads w. 7 edits	1.29	4.7	-3.41	1.13	0.4	0.73
21	% Reads w. 8 edits	1.03	3.1	-2.07	0.90	0.4	0.5
22	% Reads w. 9 edits	0.89	2.2	-1.31	0.77	0.3	0.47
23	% Reads w. 10+ edits	11.5	10.5	1.0	10.83	3.9	6.93

The last software with results is Trowel. The metrics used for testing are sensitivity, specificity, precision, and gain. These tests are done for measuring bases and for measuring reads making it eight metrics for each dataset. There are eight total datasets that we tested spread across three different organisms. The results displayed below are one table from each organism:

Dataset : SRR018292 (D. melanogaster)

Search:

	Stat	Value	Value Reported	Difference
1	No. Reads	24483260	24483260	0
2	Sensitivity (Bases)	13.043591977	66.20	-53.16
3	Specificity (Bases)	71.824486571	96.12	-24.3
4	Precision (Bases)	1.840699299	89.66	-87.82
5	Gain (Bases)	-682.534369697	58.57	-741.1
6	Sensitivity (Reads)	2.74975036	94.44	-91.69
7	Specificity (Reads)	31.20392985	99.86	-68.66
8	Precision (Reads)	0.15060123	99.67	-99.52
9	Gain (Reads)	-1820.34900635	94.13	-1914.48

Showing 1 to 9 of 9 entries

Dataset : SRR001665 (E. coli)

Search:

	Stat	Value	Value Reported	Difference
1	No. Reads	20816448	20816448	0
2	Sensitivity (Bases)	60.7666240468	78.94	-18.17
3	Specificity (Bases)	2.24283226859	99.87	-97.63
4	Precision (Bases)	0.528448764452	97.01	-96.48
5	Gain (Bases)	-11377.5236396	76.51	-11454.03
6	Sensitivity (Reads)	7.03867448971	88.82	-81.78
7	Specificity (Reads)	98.3262329334	100.0	-1.67
8	Precision (Reads)	4.71357459678	100.0	-95.29
9	Gain (Reads)	-135.25039253	88.82	-224.07

Dataset : SRR352384 (S. cerevisiae)

Search:

	Stat	Value	Value Reported	Difference
1	No. Reads	52061664	52061664	0
2	Sensitivity (Bases)	13.6201457723	15.48	-1.86
3	Specificity (Bases)	5.3168170813	99.94	-94.62
4	Precision (Bases)	0.29088511623	98.51	-98.22
5	Gain (Bases)	-4655.07070079	15.25	-4670.32
6	Sensitivity (Reads)	0.0503741835759	37.00	-36.95
7	Specificity (Reads)	96.4149975438	99.99	-3.58
8	Precision (Reads)	0.0915766058836	99.98	-99.89
9	Gain (Reads)	-54.9069504768	36.99	-91.9

Discussion

For this particular project, one cannot definitively say whether or not as a whole the project was successful. It must be broken down into each individual software and how accurate the results for each software are. In the case of Racer, the reported values for Read Search are very similar to the one that were computed by the researchers. Most of the difference values were single digit differences. On the other hand, differences increase when looking at the BWA values.

The next software is Lighter. Lighter is measured with percent increase and all the differences are less than 1. This might seem small but considering the actual metric values are all such small values, these differences are larger than the ones the researchers expected.

The software Blue features three datasets to analyze but not one of the differences is 0. In other words, not one of the number of edits that were reported by the researchers is identical to the number of edits reported by the author of the Blue paper.

The last software that the results are there for is Trowel. The values calculated for Trowel are the ones most skewed away from the ones reported. The values computed for gain are almost exponentially larger than the ones reported sometime. In other cases, the paper reports itself fixing a certain number of errors, while the values that are computed showing errors being made after the algorithm is run.

Overall, as previously mentioned, it very difficult to determine the success of this experiment. The researchers were hoping at least some of the metrics would be identical to the ones reported but that did not happen in any dataset of any software. There are several other

software that still have to be tested so it is possible that there will be changes in those. It is also hard to determine why there are such large differences. At the end of the day, the researchers are using the same tools that the authors of the paper did, and they are still seeing such large discrepancies. Potential future ideas would be to discuss methodology with the authors and developers of the software and try to determine what is the reason for the differences.

References

1. Heo, Yun, et al. "Bless: Bloom Filter-Based Error Correction Solution for High-Throughput Sequencing Reads." Bioinformatics 30.10 (2014): 1354-62. Print.
2. Ilie, Lucian, and Michael Molnar. "Racer: Rapid and Accurate Correction of Errors in Reads." Bioinformatics 29.19 (2013): 2490-93. Print.
3. Kelley, David R., Michael C. Schatz, and Steven L. Salzberg. "Quake: Quality-Aware Detection and Correction of Sequencing Errors." Genome Biology 11.11 (2010): R116. Print.
4. Lim, Eun-Cheon, et al. "Trowel: A Fast and Accurate Error Correction Module for Illumina Sequencing Reads." Bioinformatics 30.22 (2014): 3264-65. Print.
5. Greenfield, Paul, et al. "Blue: Correcting Sequencing Errors Using Consensus and Context." *Bioinformatics* 30.19 (2014): 2723-32. Print.
6. Schröder, Jan, et al. "Shrec: A Short-Read Error Correction Method." *Bioinformatics* 25.17 (2009): 2157-63. Print.
7. Song, Li, Liliana Florea, and Ben Langmead. "Lighter: Fast and Memory-Efficient Sequencing Error Correction without Counting." *Genome Biology* 15.11 (2014):